Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Currently Amended) A process for executing programs on a multiprocessor system having a plurality of processors, having an instruction set architecture, each of the processors being able to execute, at each processing cycle, a respective maximum number of instructions, the process comprising:

retrieving a first set of instructions as compiled instruction words having a first length and executable on a first processor of said plurality;

retrieving a second set of instructions as compiled instruction words having a second length and executable on a second processor of said plurality; and

modifying, during runtime of one of the programs, at least some of the instruction words of the first set of instructions by converting them into modified-instruction words executable on ~~a~~ the second processor of said plurality, said modification operation ~~in turn having at least one operation selected from the group of:~~ comprising:

splitting said instruction words of the first set of instructions into modified-instruction words and managing a context of execution of the modified-instruction words by the second ~~processor;~~ processor, wherein said first set of instructions and said second set of instructions have, respectively, a first and a second maximum length with said first maximum length greater than said second maximum length, the quotient between said first maximum length and said second maximum length having a given value with the possible presence of a remainder and in that the procedure comprises the operations of:

selectively modifying instructions in the first set of instructions by:

splitting said first instruction words into a number of said first modified-instruction words equal to the value of said quotient; and

in the presence of said remainder, adding to said first modified-instruction words a further modified-instruction word of length equal to said second

maximum length, said second maximum length being obtained by entering into said further first modified-instruction word a set of no-operation instructions. ~~and~~

~~entering in the modified-instruction words no-operation instructions.~~

2. (Currently Amended) The process according to Claim 1, further comprising:

~~retrieving a second set of instructions as compiled instruction words having a second length and executable on the second processor; and~~

modifying at least some of the instruction words of the second set of instructions into second modified-instruction words executable on said first processor of said plurality.

3. (Canceled)

4. (Previously Presented) The process according to Claim 1, further comprising:

encoding said instructions on a given number of bits, said number of bits having a first bit identifying a length of instruction word executable on a processor of said plurality;

associating to said given number of bits a respective appendix having a set of further bits identifying lengths of instruction words executable on different processors of said plurality;

identifying for each of said instructions a processor of said plurality designed to execute said instruction, said identified processor being able to process for each processing cycle a given length of instruction word; and

entering in the position of said first identifier bit a chosen bit between said further bits of said appendix, said chosen bit identifying the length of instruction word that can be executed by said identified processor.

5. (Previously Presented) The process according to Claim 4, further comprising the operation of erasing said respective appendix before execution of the instruction.

6. (Previously Presented) The process according to Claim 4, wherein said chosen bit is entered in the position of said first identifier bit in a step chosen from among:

decoding of the instruction in view of the execution;

re-filling of the cache associated to said identified processor; and

decompression of the instruction in view of the execution.

7. (Previously Presented) The process according to Claim 1, further comprising:

alternatively distributing the execution of instructions for programs between the processors of said plurality, said instructions being directly executable by the processors of said plurality in conditions of binary compatibility.

8. (Previously Presented) The process according to Claim 1, further comprising the operation of selectively distributing the execution of said instructions among the processors of said plurality, distributing dynamically the computational load of said processors.

9. (Previously Presented) The process according to Claim 1, further comprising the operation of selectively distributing the execution of said instructions between said processors of said plurality with the criterion of equalizing the operating frequency of the processors of said plurality.

10. (Previously Presented) The process according to Claim 1, further comprising the operation of performing a control process executed by at least one of the processors of said plurality so as to equalize its own workload with respect to the other processors of said multiprocessor system.

11. (Previously Presented) The process according to Claim 1, further comprising the operation of drawing up a table accessible by said control process, said table having items selected from the group of:

a list of processes being executed or suspended on any processor of said plurality of processors;

the progressive number thereof according to the order of activation;

the percentage of maximum power of the processor that is used by said process;

the execution time, said time, if zero, indicating that the process is temporarily suspended from being executed;

the amount of memory of the system used by the process to be able to execute the function for which it is responsible;

the maximum length of the long instruction that the VLIW processor can execute and for which it had been generated during compiling;

maximum length of the long instruction of the VLIW processor on which it is executed; and

the address of the portion of memory in which the data and the instructions are stored.

12. (Currently Amended) A multiprocessor system comprising:

a first processor having a given instruction set architecture and configured to execute programs with instruction words of a first length;

a second processor having the given instruction set architecture and configured to execute programs with instruction words of a second length shorter than the first length; and

means for converting instruction words of the first length compiled for execution on the first processor into modified instruction words of the second length executable on the second ~~processor.~~ processor, the means for converting being configured to selectively:

split the instruction words of the first length into a number of modified instruction words equal to a quotient between the first length and the second length; and

in a presence of a remainder, add to the modified instruction words a further modified instruction word of a length equal to the second length by adding to the remainder a set of no-operation instructions.

13. (Previously Presented) The multiprocessor system according to Claim 12, wherein said processors are all of the Very Long Instruction Word (VLIW) type.

14. (Previously Presented) The multiprocessor system according to Claim 12, wherein said plurality of processors comprises at least one VLIW processor and at least one superscalar processor.

15. (Currently Amended) A system comprising:

a plurality of processors coupled for receiving compiled instruction sets; and

a first processor of the plurality coupled to each of the other processors within said plurality, said first processor receiving from the other processors data representative of the workload of each of said other ~~processors;~~ processors and configured to generate an output signal from said first processor to said instruction set stream, said output signal controlling the instructions, which are sent to each of said processors based on the results of the workload measurement of said ~~processors.~~ processors, wherein the system is configured to selectively:

split instruction words of a first length executable on a selected processor in the plurality of processors into a number of modified-instruction words equal to a quotient between the first length and a second length, shorter than the first length, the modified instruction words being executable on another processor in the plurality of processors; and

in a presence of a remainder, add to the modified instruction words a further modified instruction word of a length equal to the second length by adding to the remainder a set of no-operation instructions.

16. (Original) The system according to Claim 15, wherein said workload measurement comprises power consumption of each of said processors of said plurality.

17. (Original) The system according to Claim 15, wherein said workload measurement comprises memory usage of each of said processors of said plurality.

18. (Original) The system according to Claim 15, wherein said workload measurement comprises number of operations carried out by each of said processors of said plurality.

19. (Currently Amended) A process of directing instruction sets to be executed by a plurality of processors in a system comprising:

receiving a plurality of executable instruction sets on a bus line connected to said processors;

receiving workload data at a first processor of said plurality of processors, said workload data being representative of workload of each of the processors of said plurality;

comparing the workload of each of the processors; ~~and~~

sending a signal from said first processor based on the data representative of the workload of each of the processors of said plurality to the bus line for modifying the number of executable instruction sets sent to each processor based on their respective ~~workloads.~~ workloads; and

selectively,

splitting instruction words of a first length executable on a selected processor in the plurality of processors into a number of modified-instruction words equal to a quotient between the first length and a second length, shorter than the first length, the modified instruction words being executable on another processor in the plurality of processors; and

in a presence of a remainder, adding to the modified instruction words a further modified instruction word of a length equal to the second length by adding to the remainder a set of no-operation instructions.

20. (Original) The process according to Claim 19, wherein said workload data includes data regarding power consumption of each of said processors of said plurality.

21. (Original) The system according to Claim 19, wherein said workload data includes data regarding memory usage of each of said processors of said plurality.

22. (Original) The system according to Claim 19, wherein said workload data includes data regarding the number of operations carried out by each of said processors of said plurality.

23-25. (Canceled)

26. (Previously Presented) The process according to claim 1, further comprising:

executing, by the second processor, the modified instruction words.

27. (Previously Presented) The multiprocessor system according to claim 12 wherein said means for converting instruction words of the first length compiled for execution on the first processor into modified instruction words of the second length executable on the second processor is configured to convert the instruction words during runtime.

28. (Previously Presented) The multiprocessor system of claim 12, further comprising:

means for selectively controlling execution of the modified instruction words by the second processor.

29. (Previously Presented) The system of claim 15 wherein the first processor is configured to manage a context of program execution for each of the processors in the plurality of processors and each of the processors is a very-large-instruction-word processor.

30. (Previously Presented) The system of claim 29 wherein managing a context of program execution includes tracking addresses in a memory in which data and instructions are stored.

31. (Previously Presented) The process of claim 19, further comprising:

managing a context of program execution for each of the processors in the plurality of processors.

32. (Previously Presented) The process of claim 19, further comprising:

modifying a set of compiled instructions sent to a processor in the plurality of processors by splitting instruction words in the set of compiled instructions into modified instruction words, wherein each of the plurality of processors is of a very-long-instruction-word type.

33. (Previously Presented) The process of claim 19, further comprising:

modifying a set of compiled instructions sent to a processor in the plurality of processors by adding no-operation instruction words to the instruction words in the set of compiled instructions.